Cryptography Overview

John Mitchell

Cryptography

- Is
  - A tremendous tool
  - The basis for many security mechanisms
- Is not
  - The solution to all security problems
  - Reliable unless implemented properly
  - Reliable unless used improperly

Basic Concepts in Cryptography

- Encryption scheme:
  - functions to encrypt, decrypt data
  - key generation algorithm
- Secret vs. public key
  - Public key: publishing *key* does not reveal key^{-1}
  - Secret key: more efficient; can have $key = key^{-1}$
- Hash function
  - Map input to short hash; ideally, no collisions
- Signature scheme
  - Functions to sign data, verify signature

Five-Minute University

Father Guido Sarducci

- Everything you could remember, five years after taking CS255 ... ?

Cryptosystem

- A cryptosystem consists of five parts
  - A set P of plaintexts
  - A set C of ciphertexts
  - A set K of keys
  - A pair of functions

    encrypt: $K \times P \rightarrow C$

    decrypt: $K \times C \rightarrow P$

  such that for every key $k \in K$ and plaintext $p \in P$

    decrypt(k, encrypt(k, p)) = p

OK def'n to start with, but doesn't include key generation or prob encryption.

Primitive example: shift cipher

- Shift letters using mod 26 arithmetic
  - Set P of plaintexts {a, b, c, ... , x, y, z}
  - Set C of ciphertexts {a, b, c, ... , x, y, z}
  - Set K of keys {1, 2, 3, ... , 25}
  - Encryption and decryption functions

    encrypt(key, letter) = letter + key (mod 26)

    decrypt(key, letter) = letter - key (mod 26)
- Example

  encrypt(3, stanford) = vwdqirug

ROT-13 is used in newsgroup postings, etc.

Evaluation of shift cipher

- Advantages
  - Easy to encrypt, decrypt
  - Ciphertext does look garbled
- Disadvantages
  - Not very good for long sequences of English words
    - Few keys -- only 26 possibilities
    - Regular pattern
      - encrypt(key,x) is same for all occurrences of letter x
      - can use letter-frequency tables, etc

Letter frequency in English

- Five frequency groups [Beker and Piper]

E has probability	0.12
TAOINSHR have probability	0.06 - 0.09
DL have probability	~ 0.04
CUMWFGYPB have probability	0.015 - 0.028
VKJXQZ have probability	< 0.01

Possible to break letter-to-letter substitution ciphers.

- 1400: Arabs did careful analysis of words in Koran
- 1500: realized that letter-frequency could break substitution ciphers

One-time pad

- Secret-key encryption scheme (symmetric)
  - Encrypt plaintext by xor with sequence of bits
  - Decrypt ciphertext by xor with same bit sequence
- Scheme for pad of length n
  - Set P of plaintexts: all n-bit sequences
  - Set C of ciphertexts: all n-bit sequences
  - Set K of keys: all n-bit sequences
  - Encryption and decryption functions

encrypt(key, text) = key $\oplus$ text (bit-by-bit)
decrypt(key, text) = key $\oplus$ text (bit-by-bit)

Evaluation of one-time pad

- Advantages
  - Easy to compute encrypt, decrypt from key, text
  - As hard to break as possible
    - This is an information-theoretically secure cipher
    - Given ciphertext, all possible plaintexts are equally likely, assuming that key is chosen randomly
- Disadvantage
  - Key is as long as the plaintext
    - How does sender get key to receiver securely?

Idea for stream cipher: use pseudo-random generators for key…

What is a "secure" cryptosystem?

- Idea
  - If enemy intercepts ciphertext, cannot recover plaintext
- Issues in making this precise
  - What else might your enemy know?
    - The kind of encryption function you are using
    - Some plaintext-ciphertext pairs from last year
    - Some information about how you choose keys
  - What do we mean by "cannot recover plaintext" ?
    - Ciphertext contains no information about plaintext
    - No efficient computation could make a reasonable guess

In practice ...

- Information-theoretic security is possible
  - Shift cipher, one-time pad are info-secure for short message
- But not practical
  - Long keys needed for good security
  - No public-key system
- Therefore
  - Cryptosystems in use are either
    - Just found to be hard to crack, or
    - Based on computational notion of security

Example cryptosystems

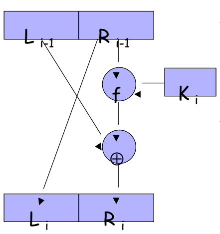
- Feistel constructions
  - Iterate a "scrambling function"
  - Example: DES, ...
  - AES (Rijndael) is also block cipher, but different
- Complexity-based cryptography
  - Multiplication, exponentiation are "one-way" fctns
  - Examples: RSA, El Gamal, elliptic curve systems, ...

Feistel networks

- Many block algorithms are *Feistel networks*
  - Examples
    - DES, Lucifer, FREAL, Khufu, Khafre, LOKI, GOST, CAST, Blowfish, ...
  - Feistel network is a standard form for
    - Iterating a function f on parts of a message
    - Producing invertible transformation
- AES (Rijndael) is related
  - also a block cipher with repeated rounds
  - not a Feistel network

Feistel network: One Round

Divide n-bit input in half and repeat



- Scheme requires
  - Function $f(R_{i-1}, K_i)$
  - Computation for K_i
    - e.g., permutation of key K
- Advantage
  - Systematic calculation
    - Easy if f is table, etc.
  - Invertible if K_i known
    - Get R_{i-1} from L_i
    - Compute $f(R_{i-1}, K_i)$
    - Compute L_{i-1} by ⊕

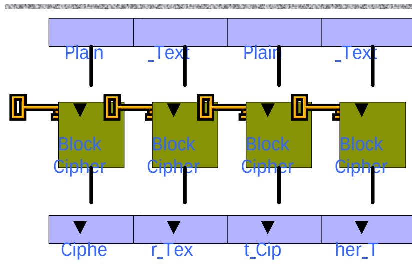
Data Encryption Standard

- Developed at IBM, widely used
- Feistel structure
  - Permute input bits
  - Repeat application of a *S-box* function
  - Apply inverse permutation to produce output
- Appears to work well in practice
  - Efficient to encrypt, decrypt
  - Not provably secure
- Improvements
  - Triple DES, AES (Rijndael)

DES modes

- ECB – Electronic Code Book mode
  - Divide plaintext into blocks
  - Encrypt each block independently, with same key
- CBC – Cipher Block Chaining
  - XOR each block with encryption of previous block
  - Use initialization vector IV for first block
- OFB – Output Feedback Mode
  - Iterate encryption of IV to produce stream cipher
- CFB – Cipher Feedback Mode
  - Output block y_i = input x_i + $\text{encyrpt}_K(y_{i-1})$

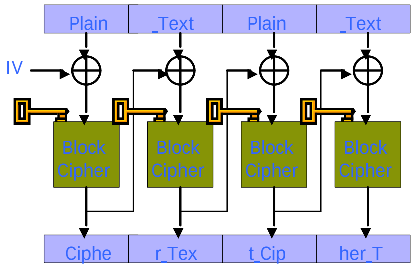
Electronic Code Book (ECB)



Problem: Identical blocks encrypted identically

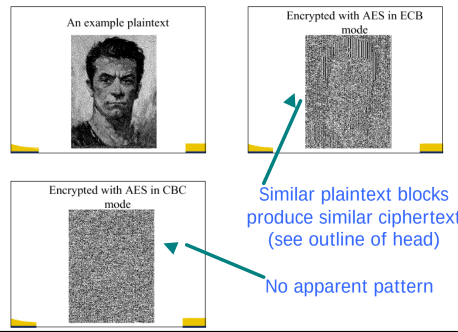
No integrity check

Cipher Block Chaining (CBC)



Advantages: Identical blocks encrypted differently
Last ciphertext block depends on entire input

Comparison (for AES, by Bart Preneel)



RC4 stream cipher – "Ron's Code"

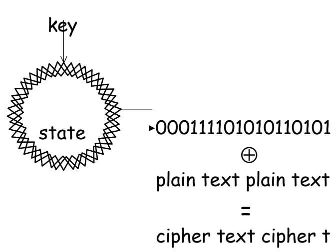
- Design goals (Ron Rivest, 1987):
  - speed
  - support of 8-bit architecture
  - simplicity (to circumvent export regulations)
- Widely used
  - SSL/TLS
  - Windows, Lotus Notes, Oracle, etc.
  - Cellular Digital Packet Data
  - OpenBSD pseudo-random number generator

RSA Trade Secret

- History
  - 1994 – leaked to cypherpunks mailing list
  - 1995 – first weakness (USENET post)
  - 1996 – appeared in Applied Crypto as "alleged RC4"
  - 1997 – first published analysis

Weakness is predictability of first bits; best to discard them

Encryption/Decryption



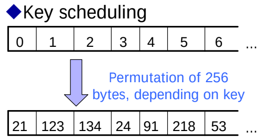
Security

- Goal: indistinguishable from random sequence
  - given part of the output stream, it is impossible to distinguish it from a random string
- Problems
  - Second byte [MS01]
    - Second byte of RC4 is 0 with twice expected probability
  - Related key attack [FMS01]
    - Bad to use many related keys (see WEP 802.11b)
- Recommendation
  - Discard the first 256 bytes of RC4 output [RSA, MS]

Complete Algorithm (all arithmetic mod 256)

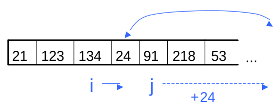
```
for i := 0 to 255    S[i] := i
j := 0
for i := 0 to 255
        j := j + S[i] + key[i]
        swap (S[i], S[j])
```

◆ Key scheduling

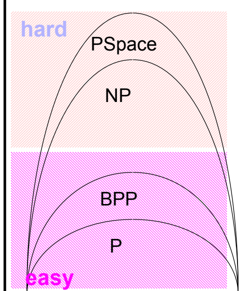



```
i, j := 0
repeat
        i := i + 1
        j := j + S[i]
        swap (S[i], S[j])
        output (S[ S[i] + S[j] ])
```

◆ Random generator



Review: Complexity Classes



- PSpace: Answer in polynomial space — may need exhaustive search
- NP: If yes, can guess and check in polynomial time
- BPP: Answer in polynomial time, with high probability
- P: Answer in polynomial time — compute answer directly

One-way functions

◆ A function f is one-way if it is
- Easy to compute f(x), given x
- Hard to compute x, given f(x), for most x

◆ Examples (we believe they are one way)
- f(x) = divide bits x = y@z and multiply f(x)=y*z
- f(x) = 3^x mod p, where p is prime
- f(x) = x^3 mod pq, where p,q are primes with |p|=|q|

One-way trapdoor

◆ A function f is *one-way trapdoor* if
- Easy to compute f(x), given x
- Hard to compute x, given f(x), for most x
- Extra “trapdoor” information makes it easy to compute x from f(x)

◆ Example (we believe)
- f(x) = x^3 mod pq, where p,q are primes with |p|=|q|
- Compute cube root using (p-1)*(q-1)

Public-key Cryptosystem

◆ Trapdoor function to encrypt and decrypt
- encrypt(key, message)

  ↕ key pair

- decrypt(key^{-1}, encrypt(key, message)) = message

◆ Resists attack
- Cannot compute m from encrypt(key, m) and key, unless you have key^{-1}

Example: RSA

◆ Arithmetic modulo pq
- Generate secret primes p, q   n
- Generate secret numbers a, b with $x^{ab} \equiv x$ mod pq

◆ Public encryption key ⟨n, a⟩
- Encrypt(⟨n, a⟩, x) = x^a mod n

◆ Private decryption key ⟨n, b⟩
- Decrypt(⟨n, b⟩, y) = y^b mod n

◆ Main properties
- This works
- Cannot compute b from n,a
  - *Apparently*, need to factor n = pq

How RSA works (quick sketch)

- Let p, q be two distinct primes and let n=p*q
  - Encryption, decryption based on group Z_n^*
  - For n=p*q, order $\phi(n)$ = (p-1)*(q-1)
    - Proof: (p-1)*(q-1) = p*q - p - q + 1
- Key pair: $\langle a, b \rangle$ with ab $\equiv$ 1 mod $\phi(n)$
  - Encrypt(x) = x^a mod n
  - Decrypt(y) = y^b mod n
  - Since ab $\equiv$ 1 mod $\phi(n)$, have $x^{ab} \equiv$ x mod n
    - Proof: if gcd(x,n) = 1, then by general group theory, otherwise use "Chinese remainder theorem".

How well does RSA work?

- Can generate modulus, keys fairly efficiently
  - Efficient rand algorithms for generating primes p,q
    - May fail, but with low probability
  - Given primes p,q easy to compute n=p*q and $\phi(n)$
  - Choose a randomly with gcd(a, $\phi(n)$)=1
  - Compute b = a-1 mod $\phi(n)$ by Euclidean algorithm
- Public key n, a does not reveal b
  - This is not proven, but believed
- But if n can be factored, all is lost ...

Public-key crypto is significantly slower than symmetric key crypto

Message integrity

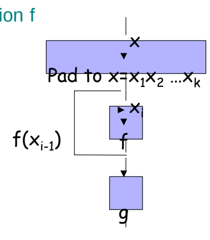
- For RSA as stated, integrity is a weak point
  - encrypt(k*m) = $(k*m)^e$ = k^e * m^e
    = encrypt(k)*encrypt(m)
  - This leads to "chosen ciphertext" form of attack
    - If someone will decrypt *new* messages, then can trick them into decrypting m by asking for decrypt(k^e *m)
- Implementations reflect this problem
  - "The PKCS#1 ... RSA encryption is intended primarily to provide confidentiality. ... It is not intended to provide integrity." RSA Lab. Bulletin
- Additional mechanisms provide integrity

One-way hash functions

- Length-reducing function h
  - Map arbitrary strings to strings of fixed length
- One way
  - Given y, hard to find x with h(x)=y
  - Given m, hard to find m' with h(m) = h(m')
- Collision resistant
  - Hard to find any distinct m, m' with h(m)=h(m')

Iterated hash functions

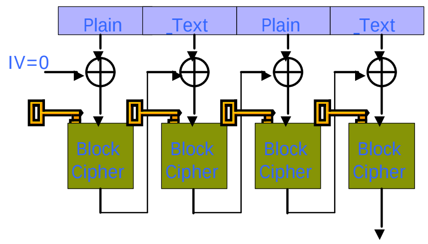
- Repeat use of block cipher or custom function
  - Pad input to some multiple of block length
  - Iterate a length-reducing function f
    - f : 2^{2k} -> 2^k reduces bits by 2
    - Repeat h_0= some seed
      h_{i+1} = f(h_i, x_i)
  - Some final function g
    completes calculation



Applications of one-way hash

- Password files (one way)
- Digital signatures (collision resistant)
  - Sign hash of message instead of entire message
- Data integrity
  - Compute and store hash of some data
  - Check later by recomputing hash and comparing
- Keyed hash fctns for message authentication
  - MAC – Message Authentication Code

Basic CBC-MAC



CBC block cipher, discarding all but last output block
Additional post-processing (e.g, encrypt with second key) can improve output

Digital Signatures

- Public-key encryption
  - Alice publishes encryption key
  - Anyone can send encrypted message
  - Only Alice can decrypt messages with this key
- Digital signature scheme
  - Alice publishes key for verifying signatures
  - Anyone can check a message signed by Alice
  - Only Alice can send signed messages

Properties of signatures

- Functions to sign and verify
  - Sign(Key^{-1}, message)
  - Verify(Key, x, m) = $\begin{cases} \text{true} & \text{if } x = \text{Sign}(Key^{-1}, m) \\ \text{false} & \text{otherwise} \end{cases}$
- Resists forgery
  - Cannot compute Sign(Key^{-1}, m) from m and Key
  - Resists existential forgery:
    given Key, cannot produce Sign(Key^{-1}, m)
    for any random or otherwise arbitrary m

RSA Signature Scheme

- Publish decryption instead of encryption key
  - Alice publishes decryption key
  - Anyone can decrypt a message encrypted by Alice
  - Only Alice can send encrypt messages
- In more detail,
  - Alice generates primes p, q and key pair $\langle a, b \rangle$
  - Sign(x) = x^a mod n
  - Verify(y) = y^b mod n
  - Since $ab \equiv 1 \bmod \phi(n)$, have $x^{ab} \equiv x \bmod n$

Public-Key Infrastructure (PKI)

- Anyone can send Bob a secret message
  - Provided they know Bob's public key
- How do we know a key belongs to Bob?
  - If imposter substitutes another key, read Bob's mail
- One solution: PKI
  - Trusted root authority (VeriSign, IBM, United Nations)
    - Everyone must know the verification key of root authority
  - Root authority can sign certificates
  - Certificates identify others, including other authorities
  - Leads to certificate chains

Crypto Summary

- Encryption scheme:
  encrypt(key, plaintext) decrypt(key^{-1},ciphertext)
- Secret vs. public key
  - Public key: publishing key does not reveal key^{-1}
  - Secret key: more efficient; can have key = key^{-1}
- Hash function
  - Map long text to short hash; ideally, no collisions
  - Keyed hash (MAC) for message authentication
- Signature scheme
  - Private key^{-1} and public key provide authentication

Limitations of cryptography

- Most security problems are not crypto problems
  - This is good
    - Cryptography works!
  - This is bad
    - People make other mistakes; crypto doesn't solve them
- Examples
  - Deployment and management problems [Anderson]
  - Ineffective use of cryptography
    - Example 802.11b WEP protocol

Why cryptosystems fail [Anderson]

- Security failures not publicized
  - Government: top secret
  - Military: top secret
  - Private companies
    - Embarrassment
    - Stock price
    - Liability
- Paper reports problems in banking industry
  - Anderson hired as consultant representing unhappy customers, 1992 class action suit

Anderson study of bank ATMs

- US Federal Reserve regulations
  - Customer not liable unless bank proves fraud
- UK regulations significantly weaker
  - Banker denial and negligence
  - Teenage girl in Ashton under Lyme
    - Convicted of stealing from her father, forced to plead guilty, later determined to be bank error
  - Sheffield police sergeant
    - Charged with theft and suspended from job; bank error
- 1992 class action suit

Sources of ATM Fraud

- Internal Fraud
  - PINs issued through branches, not post
    - Bank employees know customer's PIN numbers
  - One maintenance engineer modified an ATM
    - Recorded bank account numbers and PINs
  - One bank issues "master" cards to employees
    - Can debit cash from customer accounts
  - Bank with good security removed control to cut cost
    - No prior study of cost/benefit; no actual cost reduction
    - Increase in internal fraud at significant cost
    - Employees did not report losses to management out of fear

Sources of ATM Fraud

- External Fraud
  - Full account numbers on ATM receipts
    - Create counterfeit cards
      - Attackers observe customers, record PIN
      - Get account number from discarded receipt
    - One sys: Telephone card treated as previous bank card
      - Apparently programming bug
      - Attackers observe customer, use telephone card
  - Attackers produce fake ATMs that record PIN
  - Postal interception accounts for 30% if UK fraud
    - Nonetheless, banks have poor postal control procedures
  - Many other problems
    - Test sequence causes ATM to output 10 banknotes

Sources of ATM Fraud

- PIN number attacks on lost, stolen cards
  - Bank suggestion of how to write down PIN
    - Use weak code; easy to break
  - Programmer error - all customers issued same PIN
  - Banks store encrypted PIN on file
    - Programmer can find own encrypted PIN, look for other accounts with same encrypted PIN
  - One large bank stores encrypted PIN on mag strip
    - Possible to change account number on strip, leave encrypted PIN, withdraw money from other account

Additional problems

- Some problems with encryption products
  - Special hardware expensive; software insecure
  - Banks buy bad solutions when good ones exist
    - Not knowledgeable enough to tell the difference
  - Poor installation and operating procedures
  - Cryptanalysis possible for homegrown crypto

More sophisticated attacks described in paper

Wider Implications

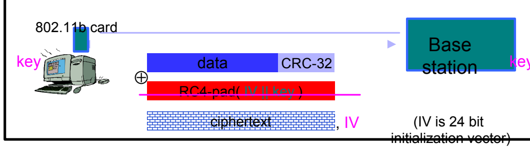
- Equipment designers and evaluators focus on technical weaknesses
  - Banking systems have some loopholes, but these do not contributed significantly to fraud
- Attacks were made possible because
  - Banks did not use products properly
  - Basic errors in
    - System design
    - Application programming
    - Administration

Summary

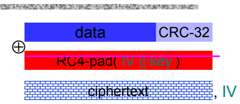
- Cryptographic systems suffer from lack of failure information
  - Understand all possible failure modes of system
  - Plan strategy to prevent each failure
  - Careful implementation of each strategy
- Most security failures due to implementation and management error
  - Program must carried out by personnel available

Last mile security: wireless Ethernet

- Many corporate wireless hubs installed without any privacy or authentication.
  - POP/IMAP passwords easily sniffed off the air.
  - Laptops in parking lot can access internal network.
- Intended “solution”: use the WEP protocol (802.11b).
  - Provides 40-bit or 128-bit encryption using RC4 ...



Some mistakes in the design of WEP



- CRC-32 ⇒ no packet integrity!!
  - CRC-32 is linear
  - Attacker can easily modify packets in transit, e.g. inject “rm –rf *”
  - Should use MAC for integrity
- Prepending IV is insufficient.
  - Fluhrer-Mantin-Shamir: RC4 is insecure in prepending IV mode
    - Given 106 packets can get key.
    - Implemented by Stubblefield, AirSnort, WEPCrack, ...
  - Correct construction:
    - packet-key = SHA-1(IV || key)
    - use longer IV, random

What to do?

- Regard 802.11b networks as public channels.
  - Use SSH, SSL, IPsec, ...
- Lesson:
  - Insist on open security reviews for upcoming standards
  - Closed standards don’t work: e.g. GSM, CMEA, ...
  - Open review worked well for SSL and IPsec

Summary

- Main functions from cryptography
  - Public-key encryption, decryption, key generation
  - Symmetric encryption
    - Block ciphers, CBC Mode
    - Stream cipher
  - Hash functions
    - Cryptographic hash
    - Keyed hash for Message Authentication Code (MAC)
  - Digital signatures
- Be careful
  - Many non-intuitive properties; prefer public review
  - Need to implement, use carefully